Computer Vision

Day 4

2017, Tanta University
Eng. Sara Hussien

Hough Transform

An edge is not a line...

How can we detect *lines* ?

Edges vs Boundaries

- Edges
  - Local intensity discontinuities
  - Points
  - Not dependent on models
- Boundaries
  - Composed of many points
  - May be dependent on model

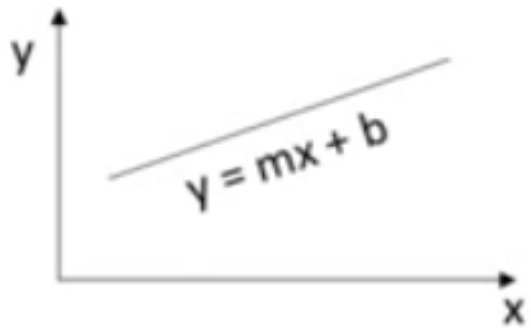
Finding lines in an image

- Option 1:
  - Search for the line at every possible position/orientation
  - What is the cost of this operation?

- Option 2:
  - Use a voting scheme: Hough transform

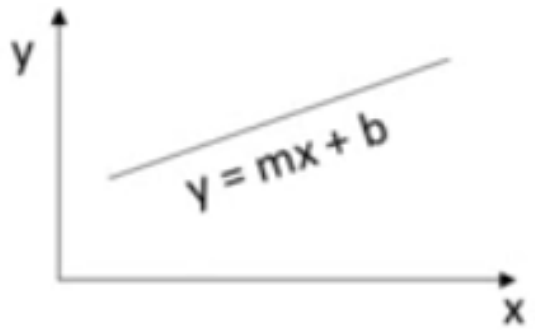
Finding lines via Hough Transform

- A line has a two parameters (m,b)



Finding lines via Hough Transform

- A line has a two parameters (m,b)

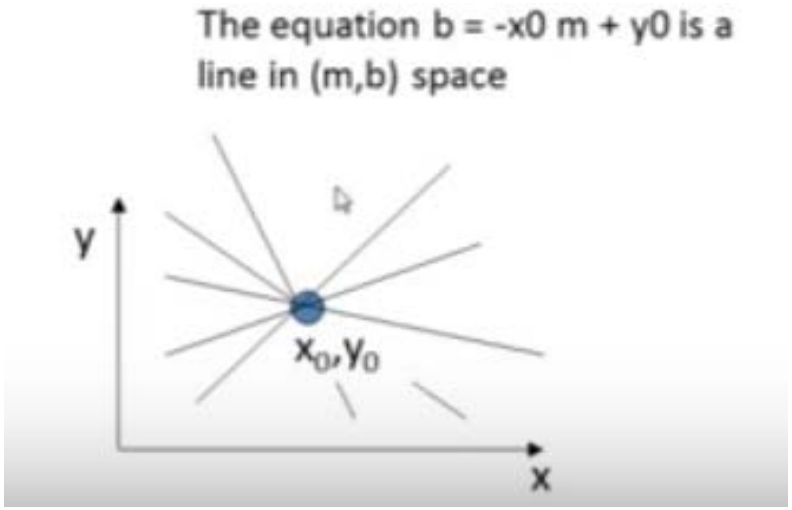


• Given a point (x0,y0), the lines that could pass through this points are all (m,b) satisfying

$$y_0 = m\,x_0 + b$$

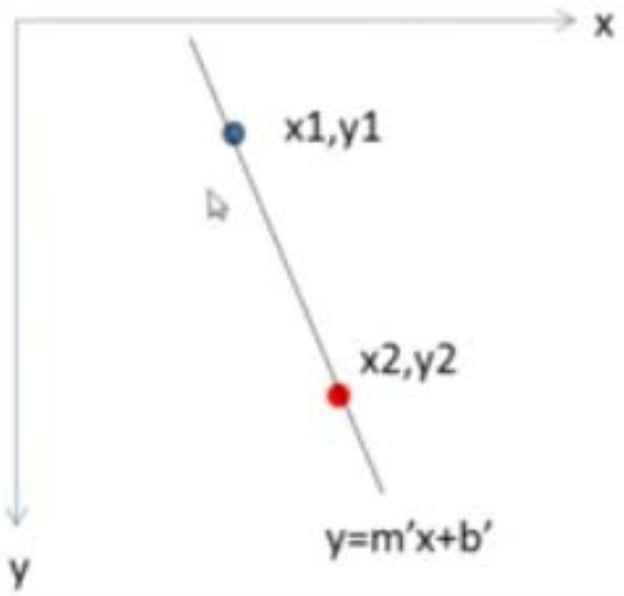
• Or

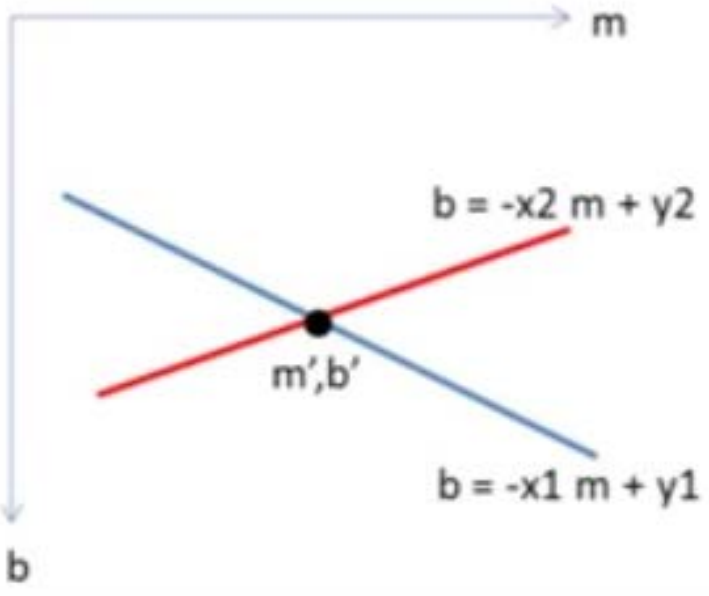
$$b = -x0\,m + y_0$$



Hough Transform (Continued)

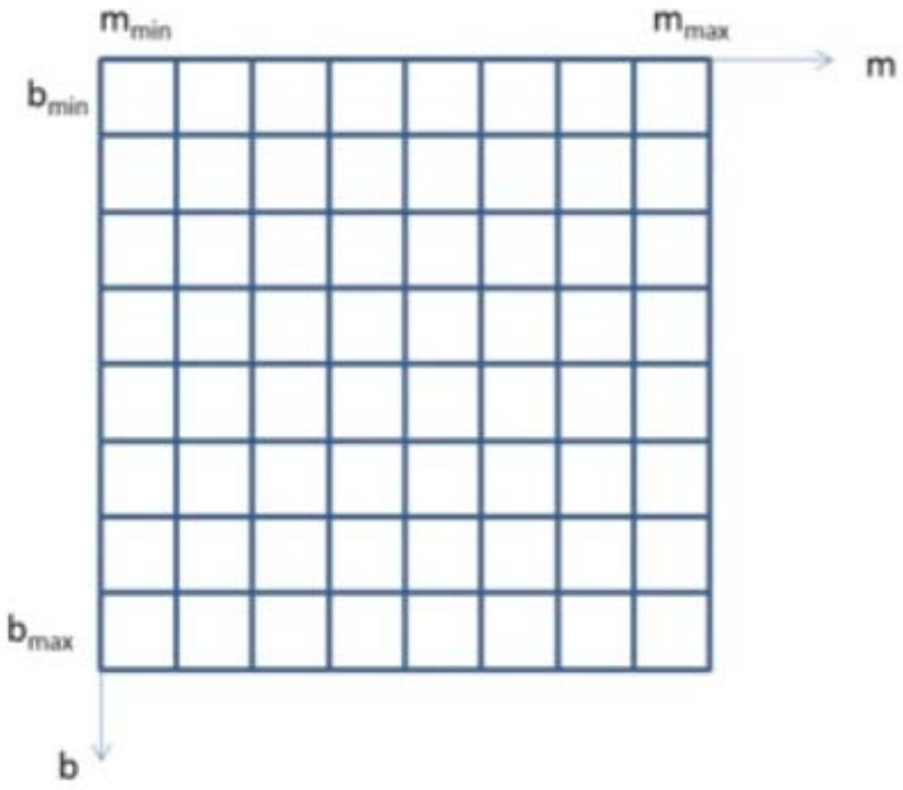
- All points on a line in image space, yield lines in parameter space which intersect at a common point
  - this point is the (m,b) of the line in image space.





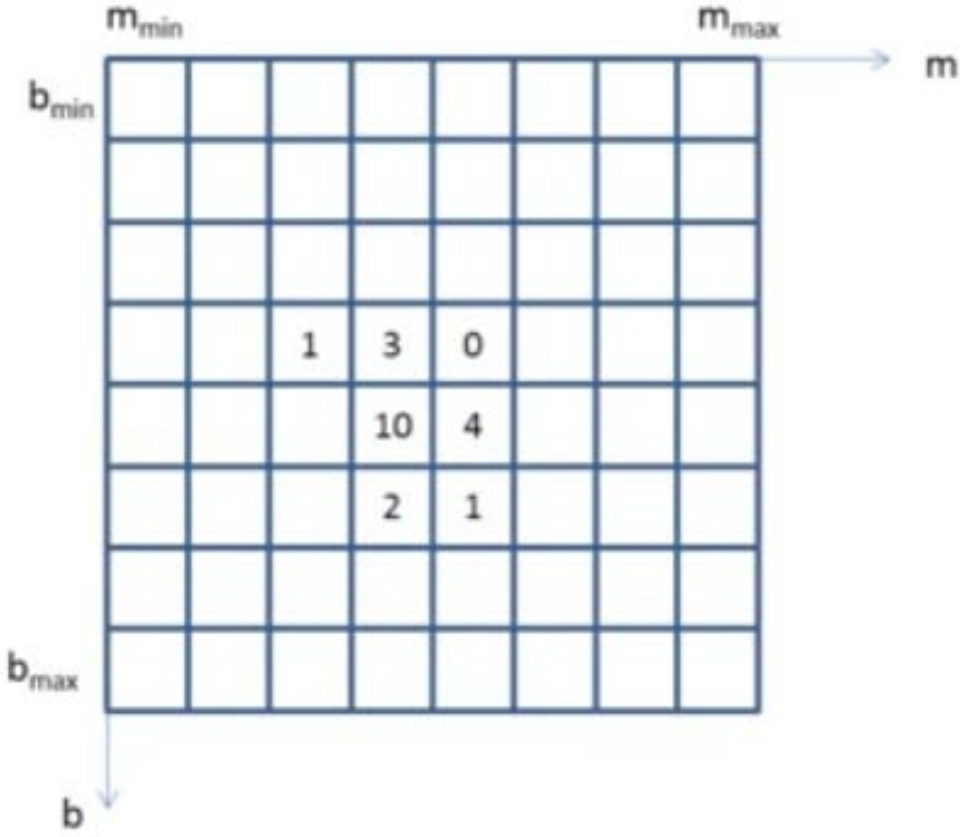
Hough Transform Algorithm

- Initialize an accumulator array A(m,b) to zero



Hough Transform Algorithm

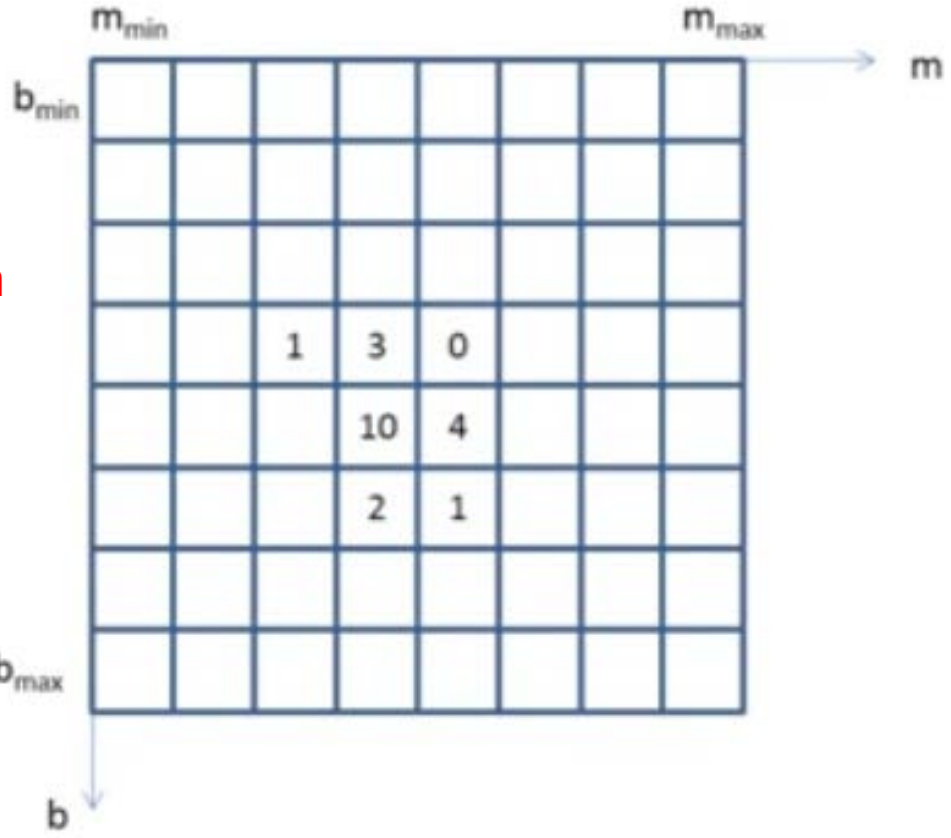
- Initialize an accumulator array A(m,b) to zero
- For each edge element (x,y), increment all cells that satisfy b = -x m + y



Hough Transform Algorithm

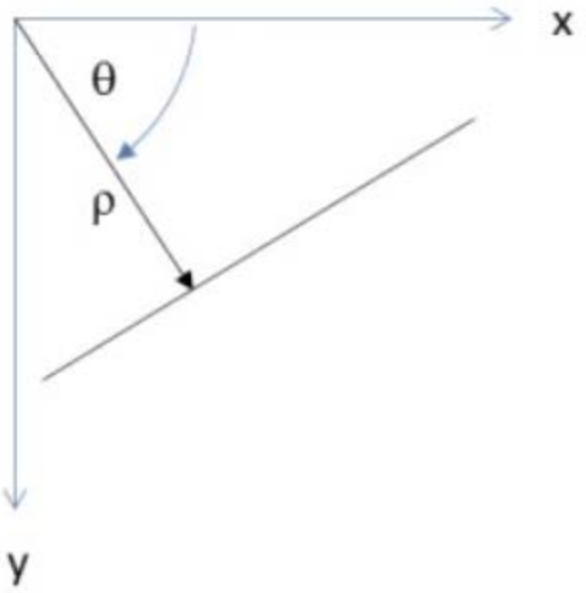
- Initialize an accumulator array A(m,b) to zero
- For each edge element (x,y), increment all cells that satisfy b = -x m + y

"This representation fails in case of vertical lines"



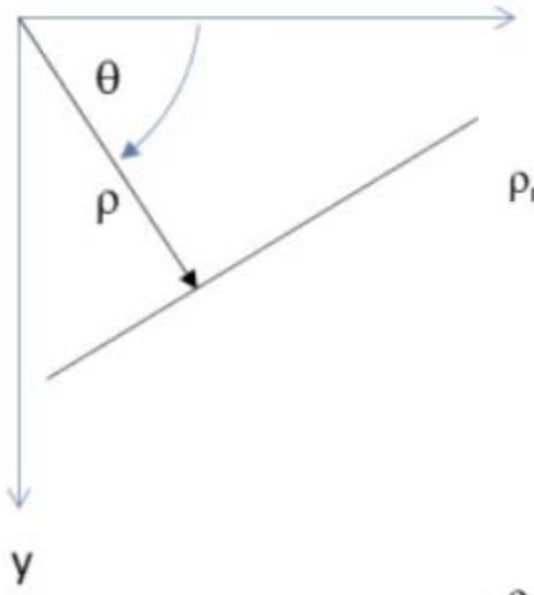
Polar Coordinate Representation of Line

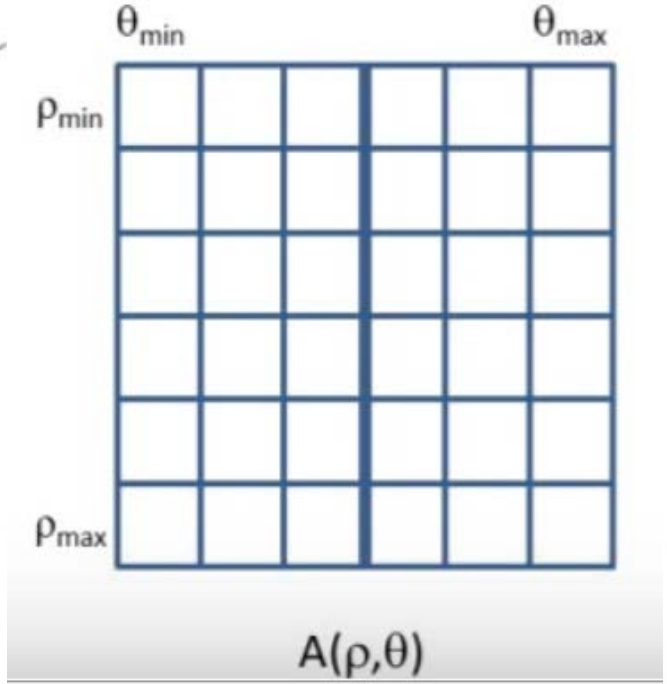
- $\rho = x \cos \theta + y \sin \theta$
  - d is the perpendicular distance from the line to the origin
  - θ is the angle this perpendicular makes with the x axis
- Avoids infinite slope



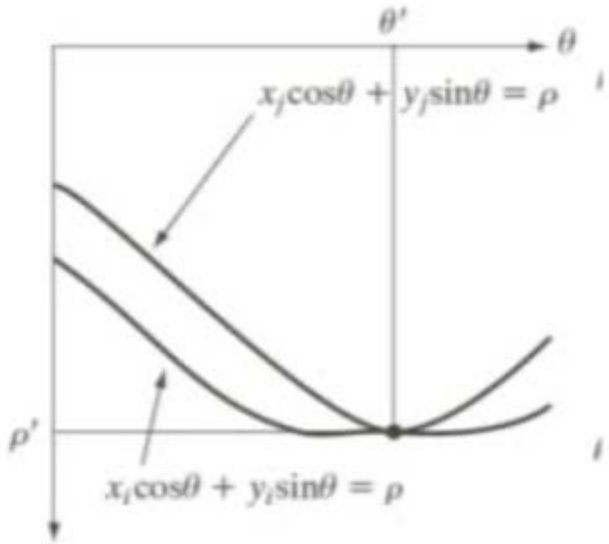
Polar Coordinate Representation of Line

- $\rho = x \cos\theta + y \sin\theta$
  - d is the perpendicular distance from the line to the origin
  - θ is the angle this perpendicular makes with the x axis
- Avoids infinite slope





The parameter space transform of a point is a sinusoidal curve



Basic Hough transform algorithm

1. Quantize the Hough Transform space: identify the maximum and minimum values of ρ and θ
2. Generate an accumulator array H[d, θ]; set all values to zero
3. for each edge point I[x,y] in the image

   for θ = 0 to 180

   $$d = xcos\theta + ysin\theta$$

   H[d, θ] += 1
4. Find the value(s) of (d, θ) where H[d, θ] is maximum
5. Calculate the equation of the line $d = xcos\theta + ysin\theta$

Extensions

- Extension 1 : Use the image gradient
  - To reduce the computational load use Gradient information

In step 2 : for each edge point I[x,y] in the image

compute unique (d, θ) based on image gradient at (x,y)

H[d, θ] += 1

- To reduce the effect of noise more than one element (elements in a neighborhood) in the accumulator array are increased

Extensions

- Extension 2 :
  - The same procedure can be used with circles, squares, or any other shape

Hough Transform for Detection of Circles

- The parametric equation of the circle can be written as

$$(x-a)^2 + (y-b)^2 = r^2$$

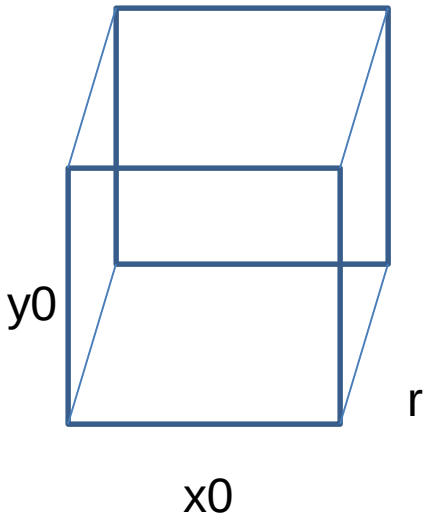
- The equation has three parameters – a, b, r
- The curve obtained in the Hough Transform space for each edge point will be a right circular cone
- Point of intersection of the cones gives the parameters a, b, r

Hough Transform for Detection of Circles

Example- circles (x0,y0,r)

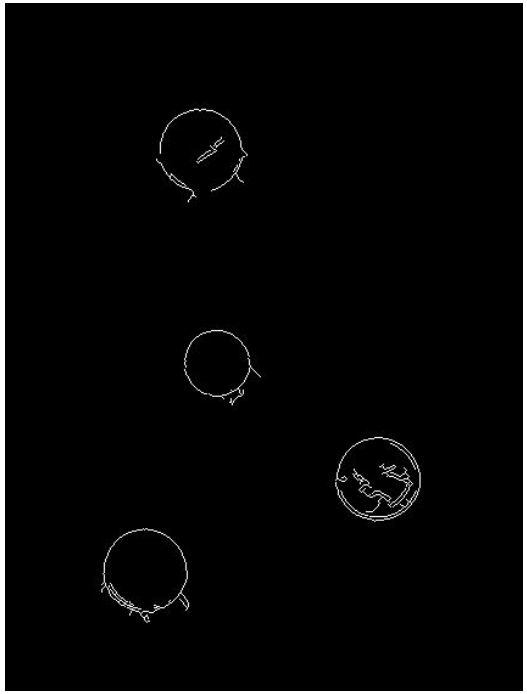
$$(x-x0)^2 + (y-y0)^2 = r^2$$

```
for all x
    for all y
        if E(x,y)
            for all x0
                for all y0
                    r=root((x-x0)^2+(y-y0)^2)
                    increment H at (x0,y0,r)
                end
            end
        end
    end
end
```



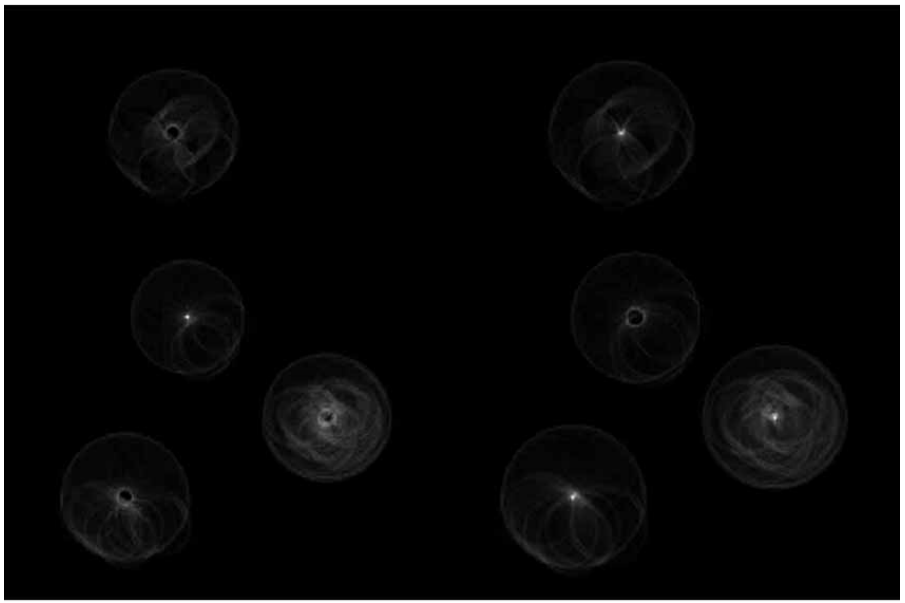
Detection of circle by Hough Transform - example

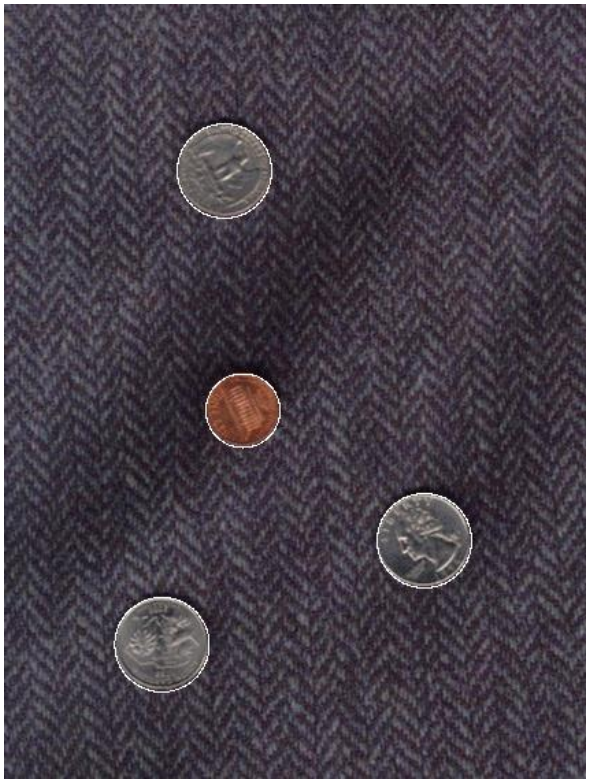
Original Image



Circles detected by Canny Edge Detector

Detection of circle by Hough Transform - contd





Hough Transform of the edge detected image

Detected Circles